Fast Fractal Image Decoding Based on Optimal Iterated Function System

Qiang Wang

College of Information Science and Technology
Dalian Maritime University
Dalian, China, 116026
Email: wangqiang2011@dlmu.edu.cn

Abstract: An optimal iterated function system (OIFS) based fast fractal decoding method was proposed in this study. Based on the minimum domain block set (MDBS) which can effectively remove redundant domain blocks and accelerate the decoding process, we found that there still exist redundant domain blocks in MDBS, and the decoding process can be accelerated further. By introducing the definition of minimum iterated function system (MIFS) and multilevel MIFSs, the redundant domain blocks can be removed level by level, and finally the optimal iterated function system (OIFS) can be obtained. Before the first iteration at decoding phase, the OIFS can be firstly established by removing the redundant domain blocks outside OIFS. Then, only the range blocks inside OIFS were reconstructed from the first to penultimate iterations, and the computations of reconstructing the range blocks outside OIFS can be saved. Finally, the whole image were reconstructed in the last iteration. Three fractal coding methods were adopted to assess the performance of the proposed method. Experimental results show that about 1%-10% of total computations in decoding process can be further saved regarding the MDBS based method.

Keywords: Fractal image coding; Fast fractal image decoding; Optimal iterated function system

1. Introduction

Fractal image coding is a successful application of fractal theory in the field of image compression. About three decades ago, after Barnsley proposed the kernel idea of fractal

image coding[1], Jacquin proposed the first practical fractal coding algorithm which has been considered as a promising image compression method[2-5]. Fractal image coding has the advantages of novel idea, potential high compression ratio, fast decoding, and resolution independence, but it suffers from high computational complexity in encoding process. Thus, to make it more suitable in practical applications, researchers worldwide proposed all kinds of fast fractal encoding methods to accelerate the encoding process: Some researchers proposed the local block matching based fast fractal encoding methods[6,7]. This type of methods tried to accelerate the encoding process by converting exhaustive block matching into local block matching in domain block pool while maintaining the decoded image quality as much as possible. To accelerate the encoding process further, some researchers proposed the no-search fractal image coding methods[8-11]. This type of methods directly appointed the best-matched domain block without block matching operations and can realize real-time encoding at the expense of poor decoded image quality. In recent years, some important improvements about fractal image coding have been also made in other aspects. For example, the block-matching operations can be simplified with fast affine transformations to accelerate the encoding process[12,13]. Linear affine transformations can be replaced with nonlinear ones to achieve better decoded image quality[14]. Several hybrid coding methods were also proposed to make fractal image coding achieve better performance[15,16]. After about three decades' development, fractal image coding has been also gradually implemented in other image processing fields, such as image retrieval[17-19], watermarking[20-22], image magnification[23-27], medical image processing[28], image hashing[29,30], human pose estimation[31,32], image denoising[33-41], and image encryption[42-44].

Although Fractal image decoding can complete the decoding process only in a small number of iterations, accelerating decoding process can also makes it more useful in real-time decoding applications. To shorten the fractal decoding process, some researchers adopt appropriate initial images which can approach the input image as much as possible, and then the initial image can easily converge to the decoded image with the specified iteration method[45-47]. Other researchers adopt better iteration strategies which can effectively reduce the iteration errors regarding the final decoded image, and thus the decoding process can be also effectively accelerated[48,49]. In recent years, completely different from the previous methods, a minimum domain block set (MDBS) based method was proposed to reduce the computational complexity in fractal decoding process[50]. MDBS contains the necessary domain blocks that can reconstruct the range blocks both inside and outside MDBS. Thus, only the range blocks inside MDBS were reconstructed in each of the first to penultimate iterations at decoding phase, and the computations of reconstructing the ones outside MDBS can be saved to speedup the decoding process.

In this study, we found that there exist redundant domain blocks as well in MDBS, and thus there is still potential to accelerate the decoding process further. We firstly introduce the definition of minimum iterated function system (MIFS) and multilevel MIFS. Then, the redundant domain blocks can be further removed level by level, and the MIFS in the last level is defined as the optimal iterated function system (OIFS). Before the first iteration at decoding phase, the OIFS can be firstly established. Then, in each of the first to penultimate iterations, only the range blocks inside OIFS are reconstructed, and the computations of reconstructing the range blocks outside OIFS can be saved. In the last iteration, the local

image inside OIFS converges, and then the range blocks outside OIFS are reconstructed as well. In summary, Compared with the MDBS based fast decoding method, the computations of reconstructing the range blocks outside OIFS and inside MIFS from the first to penultimate iterations can be saved to further speedup the fractal decoding process. By combing with the existing fast fractal decoding methods, two state-of-the-art and Jacquin's methods are adopted to assess the performance of the proposed method. Experimental results show that the proposed method can effectively reduce the number of computations in decoding process regarding the previous method. The contributions of this study can be described as:

1. Based on the definition of minimum iterated function system (MIFS), we found that there still exist redundant domain blocks inside MIFS, and thus we introduce multilevel MIFS. By removing the redundant domain blocks level by level, the OIFS can be finally obtained.
2. An OIFS based method was proposed to accelerate fractal decoding process.
3. The proposed method can be combined with existing fractal coding methods to speedup their decoding process and make them more suitable in real-time decoding applications.

This paper is organized as follows: Conventional fractal image coding is reviewed in Section 2. The definitions of MIFS, multilevel MIFS, and OIFS were introduced, and then the OIFS based fast decoding method are described in Section 3. The experimental results are presented and analyzed in Sections 4 and 5, respectively. The conclusion is given in Section 6.

2. Related work

2.1 Traditional fractal image coding

Traditional fractal image coding aims to establish an iterated function system (IFS) whose fixed point can approximate the input image well. In encoding process, the input image is first divided into range blocks with the sizes of $B\times B$, $\boldsymbol{R}_i, i=1,2,3,...,NumR$. Then, the domain block pool, $\boldsymbol{D}_j, j=1,2,3,...,NumD$, can be obtained by sliding a $2B\times 2B$ window over the input image. For each range block, the best-matched domain block is obtained by exhaustively searching within the domain block pool which is obtained by performing eight isometric transformations on it. Finally, the collage error of arbitrary range block can be computed as:

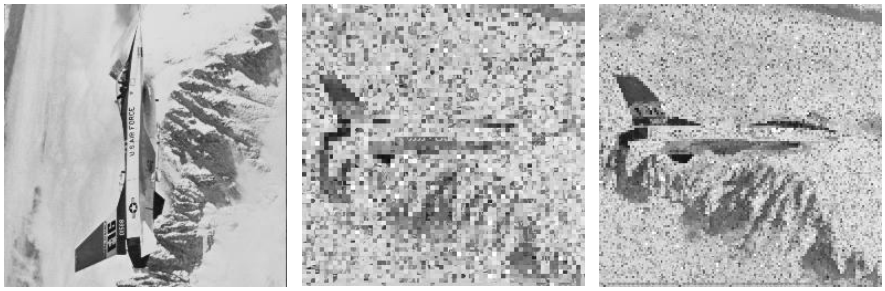
$$\mathrm{CE}(\boldsymbol{R}_i)=\underset{\boldsymbol{D}_j}{\operatorname{argmin}}\left\|\boldsymbol{R}_i-\varphi_{ij}(\boldsymbol{D}_j)\right\|=\underset{\boldsymbol{D}_j}{\operatorname{argmin}}\left\|\boldsymbol{R}_i-\left(\alpha_i\gamma_i(\eta_i(\boldsymbol{D}_j))+\beta_i\boldsymbol{I}\right)\right\| \quad (1)$$
$$i=1,2,3,\ldots,NumR,\ j=1,2,3,\ldots,NumD$$

where $\varphi_{ij}(\cdot)$ denotes the mapping operation from $\boldsymbol{D}_j$ to $\boldsymbol{R}_i$, $\eta_i(\cdot)$ and $\gamma_i(\cdot)$ denote the contracting and isometric transformations, respectively, and α_i and β_i denote the scaling and offset coefficients of the affine transformation, respectively. $\boldsymbol{I}$ denotes the $B\times B$ block whose components are all ones. From the encoding process, we know that fractal image encoding aims to establish the mapping operations from domain blocks to range blocks. All domain and range blocks, and the associated mapping operations between them constitute an IFS as:

$$\mathrm{IFS}:\left[\left(\boldsymbol{R}_i,\boldsymbol{D}_j\right),\varphi_{ij}\right], i=1,2,3,...,\mathrm{NumR},\ j=1,2,3,...,\mathrm{NumD} \quad (2)$$

In decoding process, arbitrary $M\times N$ image can be selected as the initial image, and then the decoding process can converge to the decoded image after several iterations with the same IFS in encoding process. If the Airplane image is encoded in encoding process, Figures 1 (b)-(f) illustrate the first five iteration images in decoding process while the rotated Airplane image in Fig. 1(a) was selected as the initial image. We can see that the iteration

images can gradually converge to the decoded image which can approximate the input image well.



(a) Initial Lake image (b) First iteration (c) Second iteration

(d) Third iteration (e) Fourth iteration (f) Fifth iteration

Fig. 1 Initial image and the first five iteration images in decoding process.

2.2 Fast fractal image decoding

The existing fast fractal decoding methods can be divided into two categories: 1. Adopting the initial images which can approximate the input image as much as possible. For example, the range-averaged image (RAI) can be defined by replacing the range blocks of the initial image with their respective averages as:

$$\mathrm{RAI} = \bigcup_{i=1}^{NumR} \boldsymbol{R}_i = \bigcup_{i=1}^{NumR} \left(\boldsymbol{I} \times \frac{1}{B^2} \sum_{s=1}^{B} \sum_{l=1}^{B} \boldsymbol{R}_i(l,s) \right) \tag{3}$$

The decoding process can be effectively shortened by adopting RAI as the initial image in decoding process[45,46]. Further, for fractal image denoising and magnification other than compression, the collage image (CI) coming from the encoding process can be used to directly speedup the decoding process further and described as[47]:

$$\mathrm{CI} = \bigcup_{i=1}^{NumR} \tilde{\boldsymbol{R}}_i = \bigcup_{i=1}^{NumR} \varphi_{ij}\left(\boldsymbol{D}_j\right) \tag{4}$$

2. Improving iteration strategies. Instead of mapping from one buffer to another in the decoding process of the conventional method, one-buffer-decoding (OBD) only adopted one buffer in decoding process, and the mapping operations were only conducted from one buffer to itself. Thus, the range blocks reconstructed ahead can help to reconstruct the following ones, and thus the decoding process can be effectively accelerated[48]. Further, a range block weighted (RBW) based fast decoding method was proposed. Firstly, the range blocks can be weighted by the following measure:

$$M_i = o_i \times \left(1 - s_i\right) \tag{5}$$

Larger measures implies that the associated range blocks can be reconstructed easier, and then they can be reconstructed ahead to help to reconstruct the following ones. Thus, the decoding process can be further accelerated[49].

In recent years, completely different from the previous methods, a minimum domain block set (MDBS) based method was proposed to reduce the computational complexity in decoding process[50]. Fractal encoding process consists of a series of mapping operations from domain blocks to range blocks. Actually, the block matching operations can be completed only with partial domain blocks which are defined as the minimum domain block set (MDBS). In other works, rather than all domain blocks, only the ones that can provide mapping operations for range blocks more than or equal to once constitute the minimum domain block set (MDBS) as:

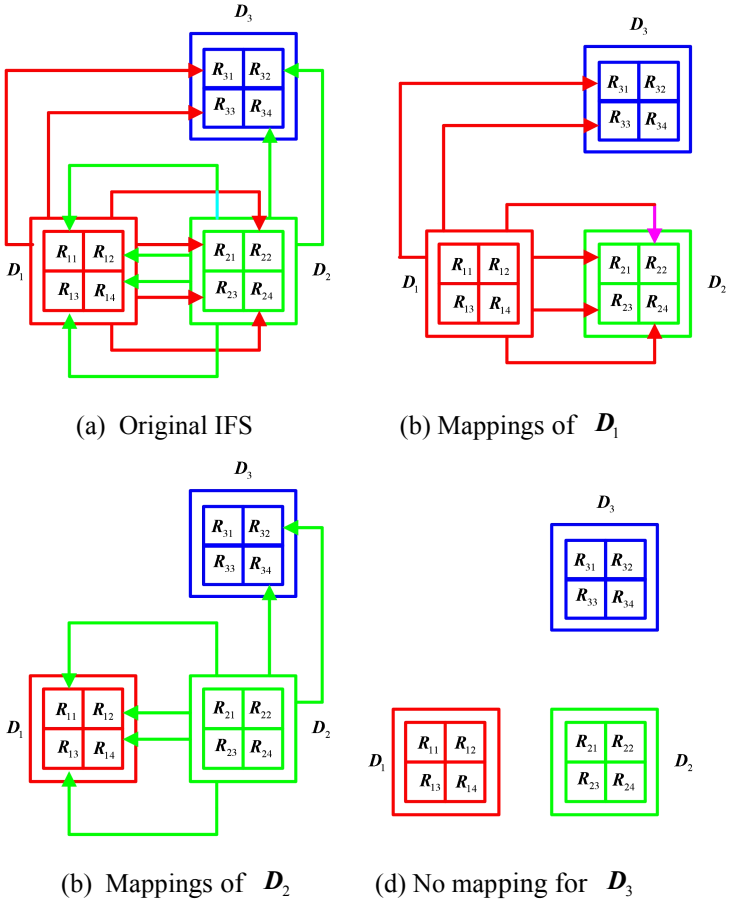
$$\mathrm{MDBS} = \bigcup_j \boldsymbol{D}_j^{\mathrm{MDBS}} \text{ s.t. } \mathrm{Times}\left(\boldsymbol{D}_j^{\mathrm{MDBS}}, \boldsymbol{R}_i\right) \geq 1, i = 1,2,3,...,\mathrm{NumR}, j = 1,2,3,...,\mathrm{NumD}_{\mathrm{MDBS}} \tag{6}$$

where Times(A, B) denotes the total number of mapping operations from A to B.

$\mathrm{NumD}_{\mathrm{MDBS}}$ denotes the total number of the domain blocks of MDBS and satisfies $\mathrm{NumD}_{\mathrm{MDBS}} \le \mathrm{NumD}$. Thus, in each of the first to penultimate iterations at decoding phase, only the range blocks inside MDBS will be reconstructed, and the computations of reconstructing the remaining range blocks outside MDBS can be saved to speedup the decoding process. In this study, we found that there are redundant domain blocks as well in MDBS. By removing the redundant domain blocks of MDBS, we can obtain the OIFS which can be adopted to speedup the decoding process further.

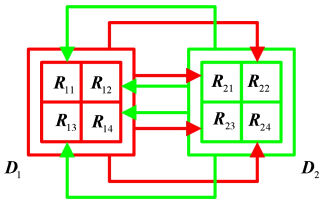
3. Accelerating fractal decoding process by optimal iterated function system

3.1 Minimum iterated function system



(a) Original IFS (b) Mappings of $\boldsymbol{D}_1$

(b) Mappings of $\boldsymbol{D}_2$ (d) No mapping for $\boldsymbol{D}_3$



(e) MIFS

Fig.2 Illustration of MIFS

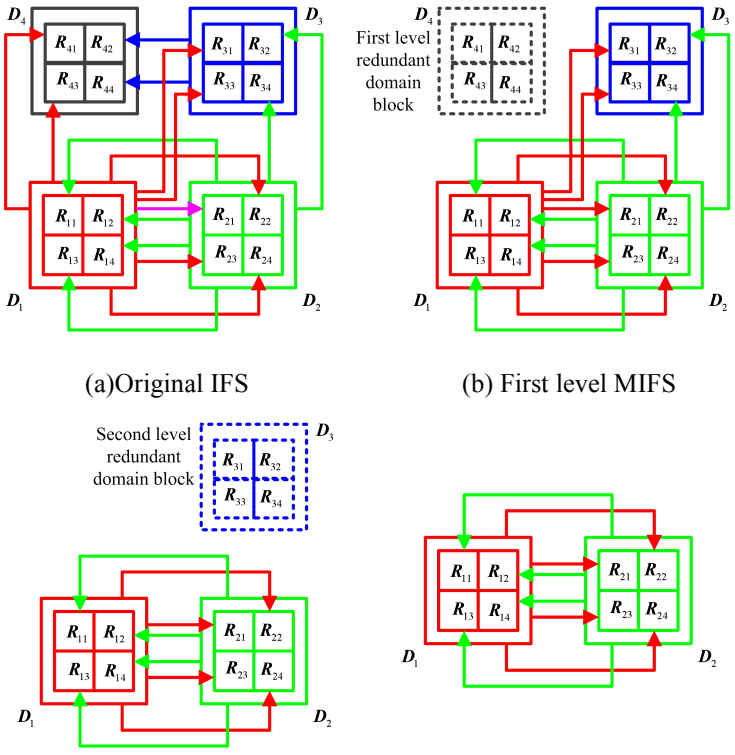
In this section, we take Fig. 2 for example to illustrate the minimum iteration function system (MIFS). For a given IFS, Fig.2 illustrates the mapping operations provided by three domain blocks for all range blocks. In Fig. 2 (a), the IFS contains three domain blocks, $\boldsymbol{D}_j, j=1,2,3$, and each of them can be uniformly divided into four range blocks, $\boldsymbol{R}_{ji}, i=1,2,3,4$. The symbol "$\rightarrow$" represents the mapping operation, $\varphi_{ij}(\cdot)$, from one domain block to another range block. Fig. 2 (b) illustrates $\boldsymbol{D}_1$ and the associated mapping operations using the red color, and then Fig. 2 (c) illustrates those for $\boldsymbol{D}_2$ using the green color. In Fig. 2 (d), $\boldsymbol{D}_3$ represented with the blue color cannot provide mapping operations for any range block. Thus, the domain blocks in Fig. 2 (a) can be divided into two categories: 1) Minimum domain block set (MDBS). This type of domain blocks consist of the domain blocks that can provide mapping operations for all range blocks. Both $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ are the domain blocks of MDBS. 2) Redundant domain block set (RDBS). This type of domain blocks consist of the domain blocks that cannot provide mapping operations for any range block. $\boldsymbol{D}_3$ belongs to the domain block of RDBS. By removing the domain block of RDBS, $\boldsymbol{D}_3$, and associated mapping operations, we can obtain the MIFS which are defined as:

Definition 1: For one IFS, the domain blocks of MDBS, range blocks inside MDBS and associated mapping operations are defined as the minimum iterated function system (MIFS) which can be described as:

$$\mathrm{MIFS}:\left[\left(\boldsymbol{R}_i^{\mathrm{MIFS}},\boldsymbol{D}_j^{\mathrm{MIFS}}\right),\varphi_{ij}\right],\ i=1,2,3,...,\mathrm{NumR}_{\mathrm{MIFS}},\ j=1,2,3,...,\mathrm{NumD}_{\mathrm{MIFS}} \tag{7}$$

where $\mathrm{NumR}_{\mathrm{MIFS}}$ denotes the total number of the range blocks inside MIFS and satisfies $\mathrm{NumR}_{\mathrm{MIFS}} \le \mathrm{NumR}$, $\mathrm{NumD}_{\mathrm{MIFS}}=\mathrm{NumD}_{\mathrm{MDBS}}$ and $\mathrm{NumR}_{\mathrm{MIFS}}=4\times\mathrm{NumD}_{\mathrm{MIFS}}$. In Fig. 2 (e), $\boldsymbol{D}_1$, $\boldsymbol{D}_2$, and the associated mapping operations between them constitute the MIFS of the original IFS in Fig. 2 (a). Thus, we know that because the domain blocks of MIFS come from those of MDBS, thus the previous MDBS based fast fractal decoding method is identical to the MIFS based fast fractal decoding method.

3.2 Multilevel Minimum iterated function systems and optimal iterated function system



(a)Original IFS (b) First level MIFS

(c) Second level MIFS (d) OIFS

Fig.3 Illustration of the MIFSs at different levels and OIFS.

In this section, we take Fig. 3 for example to illustrate the multilevel MIFSs and optimal iterated function system (OIFS). In Fig. 3 (a), we have another IFS which consists of four domain blocks, $\boldsymbol{D}_1$, $\boldsymbol{D}_2$, $\boldsymbol{D}_3$, and $\boldsymbol{D}_4$, and their associated mapping operations which are represented with the red, green, blue, and black colors, respectively. We can see that $\boldsymbol{D}_1$, $\boldsymbol{D}_2$,

and $\boldsymbol{D}_3$ in Fig. 3 (a) can provide mapping operations for some range blocks and thus belong to MDBS. But $\boldsymbol{D}_4$ cannot provide mapping operations for any range block and thus belongs to RDBS. By removing $\boldsymbol{D}_4$ and the associated mapping operations, we can obtain the MIFS in Fig. 3 (b) which is the same as the IFS in Fig. 2 (a). If the MIFS in Fig. 3 (b) is considered as a new independent IFS, we know that $\boldsymbol{D}_3$ cannot provide mapping operations for any range block and belongs to RDBS. By removing $\boldsymbol{D}_3$ and the associated mapping operations, we can obtain the MIFS in Fig. 3 (c) which is the same as that in Fig. 2 (e). We can see that if the MIFS in Fig. 3 (c) is considered as an independent IFS, there is no redundant domain block, and all domain blocks, $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$, can provide mapping operations for all range blocks inside $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$. In summary, the IFS in Fig. 3 (b) is the MIFS of the original IFS in Fig. 3 (a) and can be considered as the first level MIFS, and $\boldsymbol{D}_4$ is the first level redundant domain block. Further, The IFS in Fig. 3 (c) is the MIFS of the IFS in Fig. 3 (b) and can be considered as the second level MIFS of the original IFS in Fig. 3 (a), and $\boldsymbol{D}_3$ is the second level redundant domain block. Finally, the MIFS in Fig. 3 (c) does not have redundant domain blocks any more. By removing the redundant domain block $\boldsymbol{D}_3$ and the associated mapping operations, we can obtain the optimal iterated function system (OIFS) in Fig. 3 (d) which can be defined as:

Definition 2: For one IFS, the domain blocks that can provide mapping operations for all range blocks directly or indirectly and do not contain any redundant domain block at all, range blocks inside them, and associated mapping operations between them are defined as the optimal iterated function system (OIFS) which can be described as:

$$\mathrm{OIFS}:\left[\left(\boldsymbol{R}_i^{\mathrm{OIFS}},\boldsymbol{D}_j^{\mathrm{OIFS}}\right),\varphi_i\right],\ i=1,2,3,...,\mathrm{NumR}_{\mathrm{OIFS}},\ j=1,2,3,...,\mathrm{NumD}_{\mathrm{OIFS}} \tag{7}$$

where $\mathrm{NumR}_{\mathrm{OIFS}}$ and $\mathrm{NumD}_{\mathrm{OIFS}}$ denote the total numbers of the range blocks and domain blocks of OIFS and satisfy $\mathrm{NumR}_{\mathrm{OIFS}} \le \mathrm{NumR}_{\mathrm{MIFS}}$, $\mathrm{NumD}_{\mathrm{OIFS}} \le \mathrm{NumD}_{\mathrm{MIFS}}$, and $\mathrm{NumR}_{\mathrm{OIFS}}=4\times\mathrm{NumD}_{\mathrm{OIFS}}$. From the definition of OIFS, we know that the domain blocks of OIFS can provide the mapping operations for all range blocks inside OIFS, which implies that OIFS can reconstruct itself independently. Moreover, the domain blocks of OIFS can also provide the mapping operations for the remaining range blocks belonging to different levels of RDBSs progressively. For example, in Fig. 3, the domain blocks of OIFS in Fig. 3 (d) can progressively reconstruct the range blocks inside $\boldsymbol{D}_3$ and then the ones inside $\boldsymbol{D}_4$. Finally, the original IFS can be recovered.

3.3 Accelerating fractal decoding process with optimal iterated function system

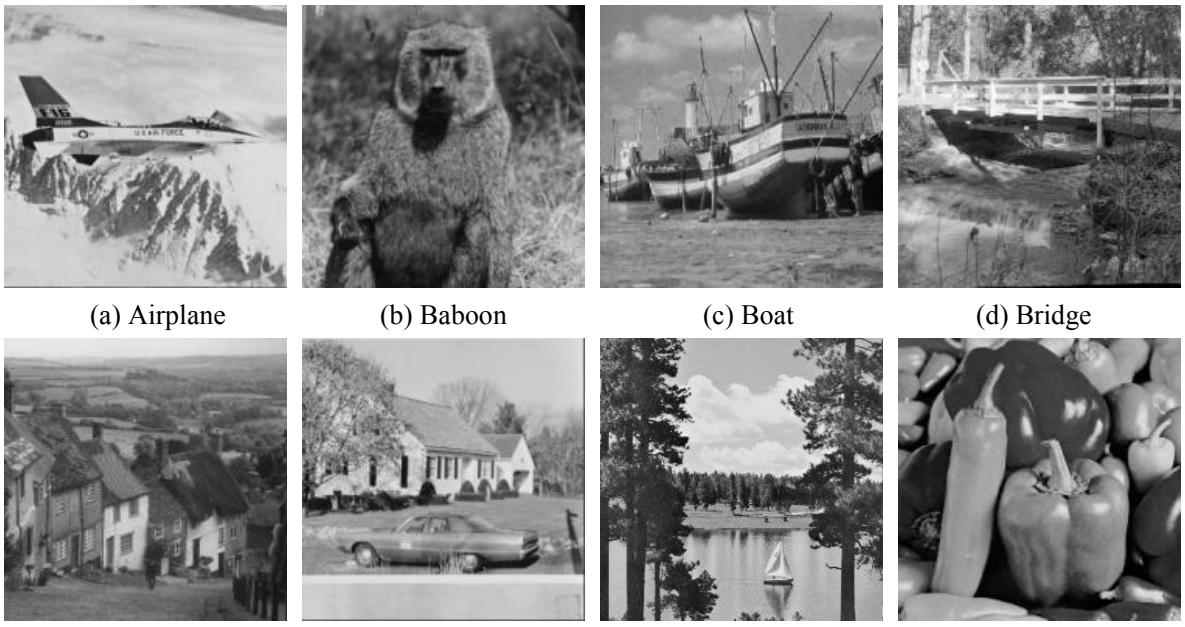
Table 1. Algorithm for calculating OIFS in decoding process.

Input: Original IFS
Output: OIFS
Initializing: Let n=0, and $\mathrm{MIFS}_{(0)}$ represents the original IFS
Check each domain block in the nth level MIFS, $\mathrm{MIFS}_{(n)}$, and determine if there exit the domain blocks belonging to the nth level RDBS, $\mathrm{RDBS}_{(n)}$
while there exit the domain blocks belonging to $\mathrm{RDBS}_{(n)}$ **do**
Remove the domain blocks belonging to $\mathrm{RDBS}_{(n)}$
Obtain the $(n+1)$th level MIFS, $\mathrm{MIFS}_{(n+1)}$
Let n=n+1
Check each domain block in the nth level $\mathrm{MDBS}_{(n)}$ and determine if there exit the domain blocks belonging to $\mathrm{RDBS}_{(n)}$
end while

Let $N=n$, and Nth level $\mathrm{MIFS}_{(N)}$ is considered as OIFS

After encoding the input image, we can have the original IFS at the decoding phase. Before the first iteration, we can obtain the OIFS with the algorithm in Table 1. Then, the decoding process can be accelerated as: In each of the first to penultimate iterations, we only reconstruct the range blocks inside OIFS, and the computations of reconstructing the range blocks outside OIFS can be saved. In the last iteration, based on OIFS, the range blocks inside MDBS were firstly reconstructed, and then the range blocks outside OIFS, i.e., the range blocks belonging to different levels of RDBSs, can be progressively reconstructed level by level in an inverse order. In summary, although the same number of computations in the last iteration are required as those in the previous method, the decoding process can be effectively accelerated by saving the computations of reconstructing the range blocks outside OIFS from the first to penultimate iterations.

4. Experiment



(a) Airplane (b) Baboon (c) Boat (d) Bridge

(e) Goldhill (f) House (g) Lake (h) Peppers

Fig. 4 Eight test images.

4.1 Experimental settings and performance metrics

In this section, eight 256×256 images, Airplane, Baboon, Boat, Bridge, Goldhill, House, Lake, and Peppers, are selected as the test images. The size of range blocks is set at 4×4, and the sliding step is 8. The scaling and offset coefficients, *s* and *o*, are quantized with 5 and 7 bits, respectively. Jacquin's method and two state-of-the-art methods, Chauraisa's[3] and Gupta's methods[4], are adopted to assess the performance of the proposed method. The root of mean square error (RMSE) is adopted to measure the deviation between the nth iteration and decoded images as:

$$\mathrm{RMSE}_n=\sqrt{\frac{1}{H\times W}\sum_{j=1}^{W}\sum_{i=1}^{H}\left[\boldsymbol{f}_{(n)}(i,j)-\boldsymbol{f}_{Decoded}(i,j)\right]^2},\ n=1,2,...,N \tag{6}$$

where $\boldsymbol{f}_{(n)}$ and $\boldsymbol{f}_{Decoded}$ denote the nth iteration and decoded images, respectively, *H* and *W* are the height and width of the images, respectively. *N* denotes the total number of iterations we need to complete the decoding process. For each of the first to penultimate iterations, only the range blocks inside OIFS are reconstructed. The ratio of $\mathrm{NumD}_{\mathrm{OIFS}}$ to NumD is adopted to describe the percentage of the computations required in each of the first to penultimate iterations of the proposed method as:

$$\mathrm{Ratio}=\frac{\mathrm{NumD}_{\mathrm{OIFS}}}{\mathrm{NumD}}\times100\% \tag{4}$$

Further, if the percentage of the computations required (PCR) for reconstructing all range blocks in each iteration is considered as 100%, the PCR of the proposed method in the whole decoding process can be calculated as:

$$\mathrm{PCR}=\frac{(N-1)\times\mathrm{Ratio}_{\mathrm{OIFS}}\times100\%+100\%}{N\times100\%}\times100\% \tag{7}$$

where the numerator and denominator represent the number of computations required in the proposed and previous methods, respectively.

4.2 Implementation details

In the experiment, the decoded image can be obtained by encoding and decoding the input image in advance, and the detailed experimental procedures are listed as:

Step 1: Select one fractal encoding method and encode the input image.

Step 2: Get OIFS by the algorithm in Table 1 before the first iteration in decoding process, and let $n=1$.

Step 3: For the nth iteration, reconstruct all the range blocks inside OIFS. Calculate RMSE_n for the local image inside OIFS. If the convergence requirement of $\mathrm{RMSE}_n \le 1$ can be satisfied, go to Step 4. If not, let $n=n+1$, turn back to Step 3, and perform another iteration.

Step 4: Reconstruct the range blocks belonging to different levels of RDBSs, and the final decoded image can be obtained.

4.3 Experimental results

In Tables 2, 3, and 4, For each test image, the second and third rows show the RMSEs and Ratios for the whole iteration images in each iteration of the previous methods. The fourth and fifth rows illustrate those for the local iteration images in the MDBS based method, and the sixth and seventh rows illustrate those for the local iteration images in the OIFS based method. "▲" represents achieving convergence in the Nth iteration. For example, for the Airplane image in Tab.2, We totally have NumD=1024 domain blocks, and OIFS contains $\mathrm{NumD}_{\mathrm{OIFS}}$=899 domain blocks which comprise $\mathrm{NumD}_{\mathrm{OIFS}}/\mathrm{NumD}\times 100\%=87.79\%$ of the input image. In each of the first to penultimate iterations, only the range blocks inside OIFS are reconstructed by the domain blocks of OIFS with the same reconstructing operations as the previous method, and thus the same RMSEs can be maintained inside OIFS in each

iteration. Then, the local image inside OIFS can achieve convergence while the convergence requirement of $\mathrm{RMSE}_{(k)} < 0.1$ inside OIFS can be satisfied. In the last iteration, the range blocks outside OIFS are also reconstructed by the domain blocks of OIFS. The convergence of the local image inside OIFS will also result in that outside OIFS, and the whole image can also achieve convergence. Thus, the same number of iterations in the proposed method can be maintained as that in the previous methods. This can be verified by the comparisons of the iteration numbers required in Tables 2, 3, and 4. Further, in each of the first to penultimate iterations, the proposed method only reconstruct the range blocks inside OIFS rather than all range blocks in Jacquin's method, and 12.21% of total computations in each iteration can be saved. In the last iteration, the range blocks both inside and outside OIFS are all reconstructed, and thus the proposed method does the same with Jacquin's method. For the whole decoding process, we know that for the Airplane image in Table 2, we have $\mathrm{NumD}_{\mathrm{MDBS}}/\mathrm{NumD} \times 100\% = 91.99\%$, and $N=3$. Thus, $\mathrm{PCR}=94.66\%$ by Eq. (7), and 5.34% of computations can be saved for the MDBS based method regarding Jacquin's method. Further, we have $\mathrm{NumD}_{\mathrm{OIFS}}/\mathrm{NumD} \times 100\% = 87.79\%$, and $N=3$. Thus, $\mathrm{PCR}=91.86\%$, and 2.80% of computations can be further saved for the OIFS based method regarding the MDBS based method.

While Jacquin's method was implemented on the eight test images, 91.99%, 85.35%, 89.65%, 87.21%, 90.14%, 88.48%, 90.04%, and 92.38% of total computations in each of the first to penultimate iterations are required for the MDBS based decoding method. Further, by Eq. (7), 94.66%, 90.23%, 92.24%, 89.34%, 93.43%, 91.36%, 92.53%, and 94.92% of total computations in the whole decoding process are required. Correspondingly, 87.79%, 77.54%,

82.91%, 81.35%, 87.50%, 85.64%, 87.30%, and 90.82% of total computations in each of the first to penultimate iterations are required for the OIFS based decoding method. Further, by Eq. (7), 91.86%, 85.03%, 87.18%, 84.46%, 91.67%, 89.23%, 90.47%, and 93.88% of total computations in the whole decoding process are required. Thus, we know that for the eight test images, 2.80%, 5.20%, 5.06%, 4.88%, 1.76%, 2.13%, 2.06%, and 1.04% of total computations can be saved for the OIFS based method regarding the MDBS based method.

Table 2 Performance comparison between Jacquin's and proposed methods in decoding process[2].

Images	Methods	Performance	Iterations (n) 1	2	3	4	5	6	PCR (%)
Airplane	Previous(Jacquin's+ OBD+RAI+RBW)	$RMSE_n$ (Whole image)	8.22	0.27	0.06▲				×
		Ratio (%)	100	100	100				100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	8.56	0.28	0.06▲				×
		Ratio (%)	91.99	91.99	100				94.66
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	8.52	0.27	0.06▲				×
		Ratio (%)	**87.79**	**87.79**	100				**91.86**
Baboon	Previous(Jacquin's+ OBD+RAI+RBW)	$RMSE_n$ (Whole image)	4.78	0.13	0.03▲				×
		Ratio (%)	100	100	100				100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	5.17	0.13	0.03▲				×
		Ratio (%)	85.35	85.35	100				90.23
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	5.17	0.14	0.03▲				×
		Ratio (%)	**77.54**	**77.54**	100				**85.03**
Boat	Previous(Jacquin's+ OBD+RAI+RBW)	$RMSE_n$ (Whole image)	7.45	0.44	0.11	0.03▲			×
		Ratio (%)	100	100	100	100			100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	7.86	0.46	0.11	0.03▲			×
		Ratio (%)	89.65	89.65	89.65	100			92.24
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	8.09	0.48	0.12	0.03▲			×
		Ratio (%)	**82.91**	**82.91**	**82.91**	100			**87.18**
Bridge	Previous(Jacquin's+ OBD+RAI+RBW)	$RMSE_n$ (Whole image)	9.11	0.76	0.32	0.17	0.10	0.07▲	×
		Ratio (%)	100	100	100	100	100	100	100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	9.75	0.80	0.34	0.18	0.11	0.07▲	×
		Ratio (%)	87.21	87.21	87.21	87.21	87.21	100	89.34
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	9.88	0.81	0.34	0.18	0.11	0.07▲	×
		Ratio (%)	**81.35**	**81.35**	**81.35**	**81.35**	**81.35**	100	**84.46**
Goldhill	Previous(Jacquin's+ OBD+RAI+RBW)	$RMSE_n$ (Whole image)	5.67	0.21	0.06▲				×
		Ratio (%)	100	100	100				100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	5.97	0.22	0.07▲				×
		Ratio (%)	90.14	90.14	100				93.43
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	6.00	0.22	0.07▲				×
		Ratio (%)	**87.50**	**87.50**	100				**91.67**
House	Previous(Jacquin's+ OBD+RAI+RBW)	$RMSE_n$ (Whole image)	7.48	0.44	0.16	0.05▲			×
		Ratio (%)	100	100	100	100			100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	7.95	0.46	0.16	0.05▲			×
		Ratio (%)	88.48	88.48	88.48	100			91.36
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	8.01	0.46	0.16	0.05▲			×
		Ratio (%)	**85.64**	**85.64**	**85.64**	100			**89.23**
Lake	Previous(Jacquin's+ OBD+RAI+RBW)	$RMSE_n$ (Whole image)	8.02	0.49	0.13	0.04▲			×
		Ratio (%)	100	100	100	100			100

	MDBS+Previous	$RMSE_n$ (Inside MDBS)	8.45	0.52	0.14	0.04▲	×
		Ratio (%)	90.04	90.04	90.04	100	92.53
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	8.47	0.52	0.14	0.04▲	×
		Ratio (%)	**87.30**	**87.30**	**87.30**	100	**90.47**
Peppers	Previous(Jacquin's+ OBD+RAI+RBW)	$RMSE_n$ (Whole image)	6.60	0.22	0.06▲		×
		Ratio (%)	100	100	100		100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	6.87	0.22	0.06▲		×
		Ratio (%)	92.38	92.38	100		94.92
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	6.91	0.23	0.06▲		×
		Ratio (%)	**90.82**	**90.82**	100		**93.88**

While Chaurasia's method was implemented on the eight test images, 81.05%, 81.15%, 81.25%, 78.81%, 81.45%, 76.86%, 79.59%, and 83.01% of total computations in each of the first to penultimate iterations are required for the MDBS based decoding method. Further, by Eq. (7), 87.37%, 87.43%, 87.50%, 84.11%, 87.63%, 82.64%, 84.69%, and 88.67% of total computations in the whole decoding process are required. Correspondingly, 73.14%, 68.55%, 70.31%, 64.55%, 68.85%, 70.21%, 70.31%, and 74.32% of total computations in each of the first to penultimate iterations are required for the OIFS based decoding method. Further, by Eq. (7), 82.09%, 79.03%, 80.21%, 73.41%, 79.23%, 77.66%, 77.73%, and 82.88% of total computations in the whole decoding process are required. Thus, we know that for the eight test images, 5.28%, 8.40%, 7.29%, 10.70%, 8.40%, 4.98%, 6.96%, and 5.79% of total computations can be saved for the OIFS based method regarding the MDBS based method.

Table 3 Performance comparison between Chaurasia's and proposed methods in decoding process[6].

Images	Methods	Performance	Iterations (n)				PCR
			1	2	3	4	(%)
Airplane	Previous(Chaurasia's +OBD+RAI+RBW)	$RMSE_n$ (Whole image)	8.26	0.22	0.04▲		×
		Ratio (%)	100	100	100		100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	9.18	0.24	0.05▲		×
		Ratio (%)	81.05	81.05	100		87.37
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	8.92	0.24	0.05▲		×
		Ratio (%)	**73.14**	**73.14**	100		**82.09**
Baboon	Previous(Chaurasia's +OBD+RAI+RBW)	$RMSE_n$ (Whole image)	4.99	0.11	0.03▲		×
		Ratio (%)	100	100	100		100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	5.54	0.13	0.03▲		×
		Ratio (%)	81.15	81.15	100		87.43
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	5.14	0.12	0.03▲		×
		Ratio (%)	**68.55**	**68.55**	100		**79.03**
Boat	Previous(Chaurasia's +OBD+RAI+RBW)	$RMSE_n$ (Whole image)	7.59	0.32	0.07▲		×
		Ratio (%)	100	100	100		100

	MDBS+Previous	$RMSE_n$ (Inside MDBS)	8.42	0.35	0.08▲		×
		Ratio (%)	81.25	81.25	100		87.50
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	8.18	0.34	0.07▲		×
		Ratio (%)	**70.31**	**70.31**	100		**80.21**
Bridge	Previous(Chaurasia's +OBD+RAI+RBW)	$RMSE_n$ (Whole image)	9.10	0.46	0.10	0.02▲	×
		Ratio (%)	100	100	100	100	100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	10.25	0.51	0.11	0.02▲	×
		Ratio (%)	78.81	78.81	78.81	100	84.11
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	9.64	0.47	0.10	0.02▲	×
		Ratio (%)	**64.55**	**64.55**	**64.55**	100	**73.41**
Goldhill	Previous(Chaurasia's +OBD+RAI+RBW)	$RMSE_n$ (Whole image)	5.89	0.22	0.05▲		×
		Ratio (%)	100	100	100		100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	6.52	0.25	0.05▲		×
		Ratio (%)	81.45	81.45	100		87.63
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	6.19	0.23	0.05▲		×
		Ratio (%)	**68.85**	**68.85**	100		**79.23**
House	Previous(Chaurasia's +OBD+RAI+RBW)	$RMSE_n$ (Whole image)	7.71	0.41	0.11	0.03▲	×
		Ratio (%)	100	100	100	100	100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	8.79	0.45	0.13	0.04▲	×
		Ratio (%)	76.86	76.86	76.86	100	82.64
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	8.56	0.44	0.13	0.04▲	×
		Ratio (%)	**70.21**	**70.21**	**70.21**	100	**77.66**
Lake	Previous(Chaurasia's +OBD+RAI+RBW)	$RMSE_n$ (Whole image)	8.22	0.61	0.17	0.05▲	×
		Ratio (%)	100	100	100	100	100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	9.21	0.68	0.19	0.05▲	×
		Ratio (%)	79.59	79.59	79.59	100	84.69
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	8.81	0.65	0.18	0.05▲	×
		Ratio (%)	**70.31**	**70.31**	**70.31**	100	**77.73**
Peppers	Previous(Chaurasia's +OBD+RAI+RBW)	$RMSE_n$ (Whole image)	6.69	0.19	0.05▲		×
		Ratio (%)	100	100	100		100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	7.35	0.21	0.05▲		×
		Ratio (%)	83.01	83.01	100		88.67
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	7.21	0.21	0.05▲		×
		Ratio (%)	**74.32**	**74.32**	100		**82.88**

While Gupta's method was implemented on the eight test images, 84.18%, 81.64%, 83.01%, 80.76%, 83.69%, 80.86%, 83.98%, and 84.18% of total computations in each of the first to penultimate iterations are required for the MDBS based decoding method. Further, by Eq. (7), 89.45%, 87.76%, 87.26%, 85.57%, 89.13%, 85.64%, 87.99%, and 89.45% of total computations in the whole decoding process are required. Correspondingly, 78.81%, 72.17%, 76.17%, 76.63%, 78.42%, 76.27%, 78.32%, and 78.91% of total computations in each of the first to penultimate iterations are required for the OIFS based decoding method. Further, by Eq. (7), 85.87%, 81.45%, 82.13%, 80.22%, 85.61%, 82.20%, 83.74%, and 85.94% of total

computations in the whole decoding process are required. Thus, we know that for the eight test images, 3.58%, 6.31%, 5.13%, 5.35%, 3.52%, 3.44%, 4.25%, and 3.51% of total computations can be saved for the OIFS based method regarding the MDBS based method.

In summary, from the above results, we know that about 1%-10% of total computations can be saved for the OIFS based method regarding the MDBS based method.

Table 4 Performance comparison between Gupta's and proposed methods in decoding process[7].

Images	Methods	Performance	Iterations (n) 1	2	3	4	PCR (%)
Airplane	Previous(Gupta's+OBD+RAI+RBW)	$RMSE_n$ (Whole image)	8.07	0.26	0.06▲		×
		Ratio (%)	100	100	100		100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	8.79	0.29	0.06▲		×
		Ratio (%)	84.18	84.18	100		89.45
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	8.56	0.28	0.06▲		×
		Ratio (%)	**78.81**	**78.81**	100		**85.87**
Baboon	Previous(Gupta's+OBD+RAI+RBW)	$RMSE_n$ (Whole image)	4.95	0.17	0.04▲		×
		Ratio (%)	100	100	100		100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	5.48	0.19	0.04▲		×
		Ratio (%)	81.64	81.64	100		87.76
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	5.16	0.17	0.04▲		×
		Ratio (%)	**72.17**	**72.17**	100		**81.45**
Boat	Previous(Gupta's+OBD+RAI+RBW)	$RMSE_n$ (Whole image)	7.49	0.47	0.12	0.03▲	×
		Ratio (%)	100	100	100	100	100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	8.22	0.51	0.13	0.03▲	×
		Ratio (%)	83.01	83.01	83.01	100	87.26
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	8.11	0.50	0.13	0.03▲	×
		Ratio (%)	**76.17**	**76.17**	**76.17**	100	**82.13**
Bridge	Previous(Gupta's+OBD+RAI+RBW)	$RMSE_n$ (Whole image)	9.26	0.59	0.14	0.03▲	×
		Ratio (%)	100	100	100	100	100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	10.29	0.65	0.15	0.03▲	×
		Ratio (%)	80.76	80.76	80.76	100	85.57
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	10.02	0.63	0.15	0.03▲	×
		Ratio (%)	**73.63**	**73.63**	**73.63**	100	**80.22**
Goldhill	Previous(Gupta's+OBD+RAI+RBW)	$RMSE_n$ (Whole image)	5.74	0.24	0.05▲		×
		Ratio (%)	100	100	100		100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	6.28	0.25	0.05▲		×
		Ratio (%)	83.69	83.69	100		89.13
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	6.17	0.25	0.05▲		×
		Ratio (%)	**78.42**	**78.42**	100		**85.61**
House	Previous(Gupta's+OBD+RAI+RBW)	$RMSE_n$ (Whole image)	7.70	0.38	0.15	0.05▲	×
		Ratio (%)	100	100	100	100	100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	8.56	0.42	0.16	0.05▲	×
		Ratio (%)	80.86	80.86	80.86	100	85.64
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	8.44	0.41	0.16	0.05▲	×
		Ratio (%)	**76.27**	**76.27**	**76.27**	100	**82.20**
Lake	Previous(Gupta's+OBD+RAI+RBW)	$RMSE_n$ (Whole image)	8.15	0.45	0.13	0.04▲	×
		Ratio (%)	100	100	100	100	100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	8.89	0.49	0.14	0.04▲	×

	OIFS+Previous	Ratio (%)	83.98	83.98	83.98	100	87.99
		$RMSE_n$ (Inside OIFS)	8.69	0.48	0.13	0.04▲	×
		Ratio (%)	**78.32**	**78.32**	**78.32**	100	**83.74**
Peppers	Previous(Gupta's+OBD+RAI+RBW)	$RMSE_n$ (Whole image)	6.36	0.21	0.06▲		×
		Ratio (%)	100	100	100		100
	MDBS+Previous	$RMSE_n$ (Inside MDBS)	6.93	0.23	0.07▲		×
		Ratio (%)	84.18	84.18	100		89.45
	OIFS+Previous	$RMSE_n$ (Inside OIFS)	6.80	0.22	0.06▲		×
		Ratio (%)	**78.91**	**78.91**	100		**85.94**

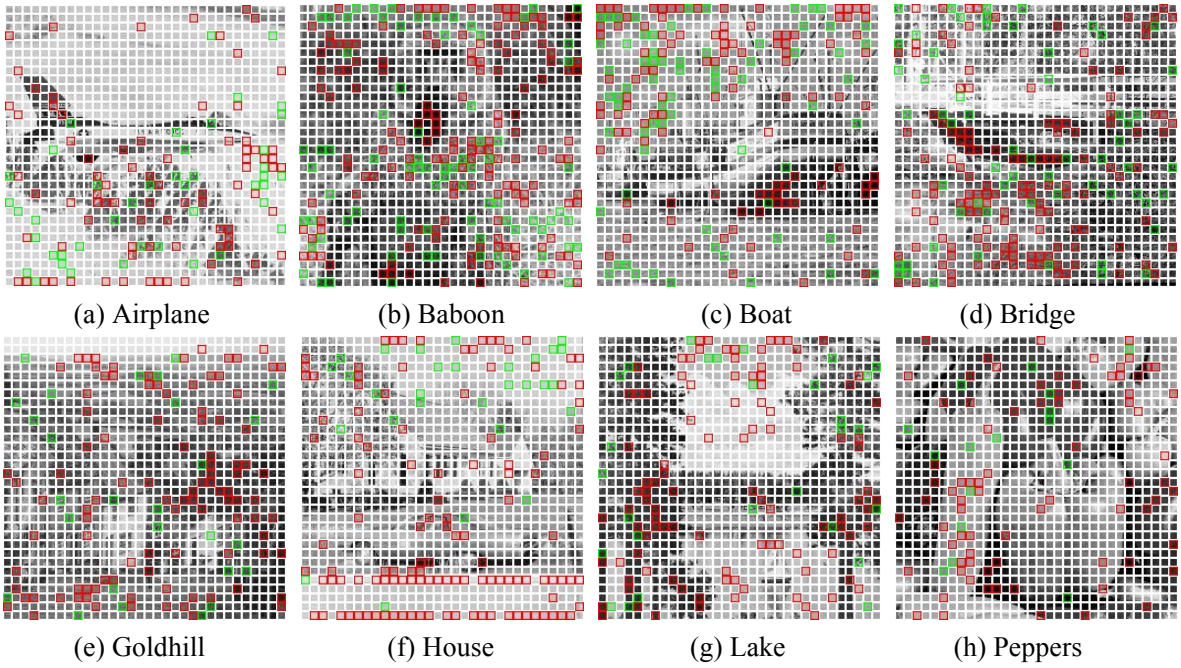
4.4 Analysis on experimental results

For Jacquin's method, Table 5 illustrates the numbers of the domain blocks from the first to last level MIFSs for the eight test images. We take the airplane image for example to compare the previous MDBS based methods with the proposed OIFS based method. We totally have 1024 domain blocks. The first to fourth level MIFSs contain 942, 914, 901, and 899 domain blocks, respectively. Then, we know that compared with the previous method, the MDBS based method can save the computations of reconstructing 1024-942=82 domain blocks. Further, the proposed OIFS based method can save the computations of reconstructing 942-899=43 domain blocks further. Similarly, the computations of reconstructing 80, 69, 60, 27, 29, 28, and 16 domain blocks can be also saved for the OIFS based method regarding the MDBS based method for the remaining seven test images. In Fig. 5, for the eight test images, both the white and green blocks represent the domain blocks of MDBS, and the white blocks alone represent the domain blocks of OIFS. In other words, the green blocks represent the domain blocks from the second to last levels of RDBSs whose computations can be saved regarding the MDBS based method.

Table 5 The numbers of domain blocks in different levels of MDBSs for Jacquin's method[2].

Images	Airplane	Baboon	Boat	Bridge	Goldhill	House	Lake	Peppers
$NumD_{MDBS}$	942	874	918	893	923	906	922	946
$NumD_{(2)}$	914	828	882	851	900	888	896	933
$NumD_{(3)}$	901	808	863	840	897	881	894	930

$NumD_{(4)}$	899	801	855	835	896	879		
$NumD_{(5)}$		796	851	833		877		
$NumD_{(6)}$		794	850					
$NumD_{(7)}$			849					
$NumD_{OIFS}$	899	794	849	833	896	877	894	930



(a) Airplane (b) Baboon (c) Boat (d) Bridge

(e) Goldhill (f) House (g) Lake (h) Peppers

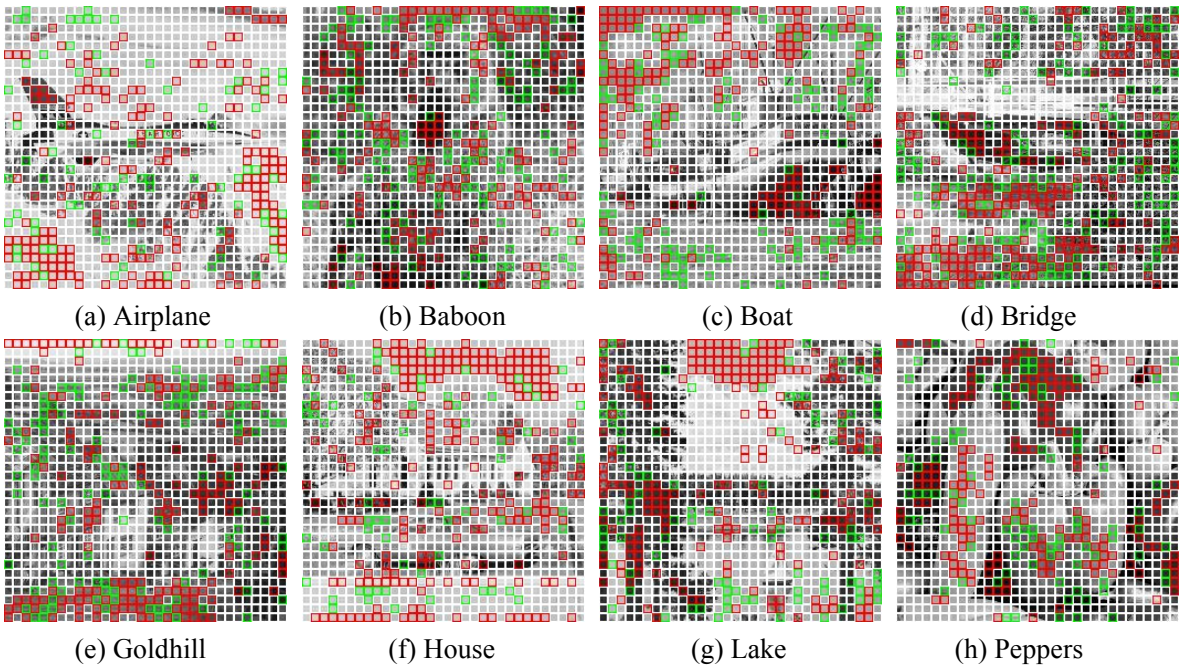
Fig. 5 Illustration of the domain blocks of MDBS and OIFS, respectively, for the eight test images.

For Chaurasia's method, Table 6 illustrates the numbers of the domain blocks from the first to last level MIFSs for the eight test images. We need to reconstruct 830, 831, 832, 807, 834, 787, 815, and 850 domain blocks for the MDBS based method in each of the first to penultimate iterations. Moreover, We need to reconstruct 749, 702, 720, 661, 706, 719, 720, and 761 domain blocks for the OIFS based method. Then, we know that the computations of reconstructing 81, 129, 112, 41, 128, 68, 95, and 89 domain blocks can be also saved for the OIFS based method regarding the MDBS based method for the eight test images. Similar with Fig. 5, both the white and green blocks in Fig. 6 represent the domain blocks of MDBS for the eight test images, and the white blocks alone represent the domain blocks of OIFS. Thus, the green blocks represent the domain blocks

from the second to last levels of RDBSs whose computations can be saved regarding the MIFS based method.

Table 6 The numbers of domain blocks in different levels of MDBSs for Chaurasia's method[6].

Images	Airplane	Baboon	Boat	Bridge	Goldhill	House	Lake	Peppers
$NumD_{MDBS}$	830	831	832	807	834	787	815	850
$NumD_{(2)}$	772	757	764	720	768	739	749	789
$NumD_{(3)}$	753	728	736	689	741	723	731	770
$NumD_{(4)}$	750	710	728	676	722	720	724	764
$NumD_{(5)}$	749	705	724	666	715	719	721	761
$NumD_{(6)}$		703	722	661	712		720	
$NumD_{(7)}$		702	721		710			
$NumD_{(8)}$			720		709			
$NumD_{(9)}$					706			
$NumD_{(10)}$					705			
$NumD_{OIFS}$	749	702	720	661	705	719	720	761



(a) Airplane (b) Baboon (c) Boat (d) Bridge

(e) Goldhill (f) House (g) Lake (h) Peppers

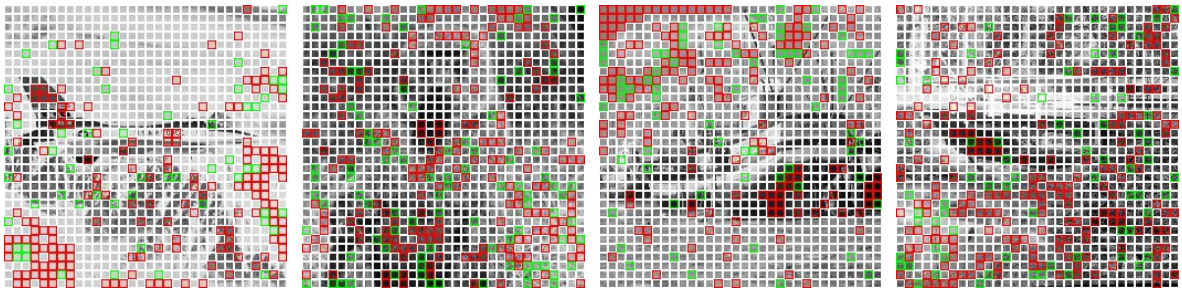
Fig. 6 Illustration of the domain blocks of MDBS and OIFS, respectively, for the eight test images.

For Gupta's method, Table 7 illustrates the numbers of the domain blocks from the first to last level MIFSs for the eight test images. We need to reconstruct 862, 836, 850, 827, 857, 828, 860, and 862 domain blocks for the MDBS based method in each of the

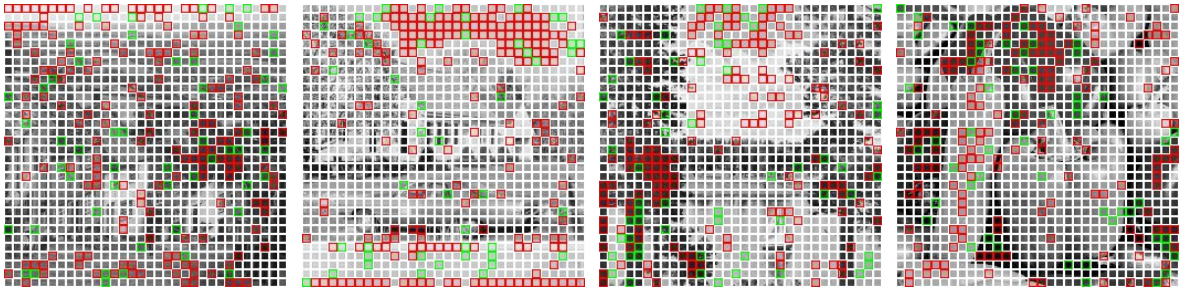
first to penultimate iterations. Moreover, We need to reconstruct 807, 739, 780, 754, 803, 781, 802, and 808 domain blocks for the OIFS based method. Then, we know that the computations of reconstructing 55, 97, 70, 73, 54, 47, 58, and 54 domain blocks can be also saved for the OIFS based method regarding the MDBS based method for the eight test images. In Fig. 7, both the white and green blocks represent the domain blocks of MDBS for the eight test images, and the white blocks alone represent the domain blocks of OIFS. Thus, the green blocks represent the domain blocks from the second to last levels of RDBSs whose computations can be saved regarding the MIFS based method.

Table 7 The numbers of domain blocks in different levels of MDBSs for Gupta's method[7].

Images	Airplane	Baboon	Boat	Bridge	Goldhill	House	Lake	Peppers
$NumD_{MDBS}$	862	836	850	827	857	828	860	862
$NumD_{(2)}$	824	778	801	774	819	797	816	813
$NumD_{(3)}$	809	754	787	756	809	787	802	809
$NumD_{(4)}$	807	745	783	754	804	783		808
$NumD_{(5)}$		742	781		803	781		
$NumD_{(6)}$		740	780					
$NumD_{(7)}$		739						
$NumD_{OIFS}$	807	739	780	754	803	781	802	808



(a) Airplane (b) Baboon (c) Boat (d) Bridge



(e) Goldhill (f) House (g) Lake (h) Peppers

Fig. 7 Illustration of the domain blocks of MDBS and OIFS, respectively, for the eight test images.

5. Conclusion

An OIFS based fast fractal decoding method is proposed in this paper. Firstly, the definitions of MIFS, multilevel MIFS, and OIFS are introduced. In decoding process, OIFS can independently achieve convergence within OIFS by the same number of iterations as before. Thus, the computations of reconstructing the range blocks outside OIFS can be saved. However, the reconstruction of all range blocks both inside and outside OIFS in the last iteration should be performed to recover the whole decoded image. Finally, Jacquin's and two state of the art methods are adopted to verify the effectiveness of the proposed method. Through reducing the computational complexity in decoding process, the proposed method can make fractal image coding more suitable for the occasions with real-time decoding requirements, such as remote monitoring, video conferencing, and virtual reality.

Conflict of Interests

The author declares that he has no conflict of interests regarding the publication of this paper.

References

[1] Barnsley, M. fractal eerywhere. Academic Press, New York: 1988.

[2] Jacquin AE. Image coding based on a fractal theory of iterated contractive image transformations. IEEE Trans Image Process 1992; 1(1): 18-30. https://doi.org/10.1109/83.128028.

[3] Fisher Y. Fractal Image Compression: Theory and Application. Springer-Verlag; 1994.

[4] Wohlberg B, Jager G de. A review of the fractal image coding literature. IEEE Trans Image Process 1999; 8(12): 1716-1729. https://doi.org/10.1109/83.806618.

[5] Jacquin AE. Fractal image coding: A review. Proc IEEE 1993; 81(10): 1451-1465. https://doi.org/10.1109/5.241507.

[6] Chaurasia V, Chaurasia V. Statistical feature extraction based technique for fast fractal image

compression. J Vis Commun Image R 2016; 41: 87-95. https://doi.org/10.1016/j.jvcir.2016.09.008.

[7] Gupta R, Mehrotra D, Tyagi RK. Hybrid edge-based fractal image encoding using K-NN search. Multimed Tools Appl 2022; 81(15): 21135-21154. https://doi.org/10.1007/s11042-022-12631-7.

[8] Shen FR, Osamu H. A fast no search fractal image coding method. Signal Process Image Commun 2004; 19(5): 393-404. https://doi.org/10.1016/j.image.2004.02.002.

[9] Wang XY, Wang SG. An improved no-search fractal image coding method based on a modified gray-level transform. Comput Graph 2008; 32(4): 445-450. https://doi.org/10.1016/j.cag.2008.02.004.

[10] Wang XY, Wang YX, Yun JJ. An improved no-search fractal image coding method based on a fitting plane. Image Vision Comput 2010; 28, 1303-1308. https://doi.org/10.1016/j.imavis.2010.01.008.

[11] Bi S, Wang Q. Fractal image coding based on a fitting surface. J Appl Math 2014; 634848. https://doi.org/10.1155/2014/634848.

[12] Nandi U. Fractal image compression using a fast affine transform and hierarchical classification scheme. The Visual Computer 2022; 38: 3867–3880. https://doi.org/10.1007/s00371-021-02226-y.

[13] Roy SK, Kumar S, Chanda B, Chaudhuri BB, Banerjee S. Fractal image compression using upper bound on scaling parameter. Chaos Solitons & Fractals 2018; 106: 16-22. https://doi.org/10.1016/j.chaos.2017.11.013.

[14] Nandi U. Fractal image compression with adaptive quadtree partitioning and non-linear affine map. Multimed Tools Appl 2020; 79: 26345–26368. https://doi.org/10.1007/s11042-020-09256-z.

[15] Nandi U. An adaptive fractal-based image coding with hierarchical classification strategy and its modifications. Innovations Syst Softw Eng 2019; 15: 35–42. https://doi.org/10.1007/s11334-019-00327-5.

[16] Wang Q, Jin GH, Bi S. Adaptively hybrid fractal image coding. IET Image Process 2024; 18(7): 1745-1758. https://doi.org/10.1049/ipr2.13060.

[17] Pi MH, Mandal MK, Basu A. Image retrieval based on histogram of fractal parameters. IEEE Trans Multimedia 2005; 7(4): 597-605. https://doi.org/10.1109/TMM.2005.846796.

[18] Wang XY, Chen Z. A fast fractal coding in application of image retrieval. Fractals 2009; 17(4): 441-450. https://doi.org/10.1142/s0218348x09004557.

[19] Huang X, Zhang Q, Liu W. A new method for image retrieval based on analyzing fractal coding characters. J Vis Commun Image R 2013; 24(1): 42-47. https://doi.org/10.1016/j.jvcir.2012.10.005.

[20] Pi MH, Li CH, Li H. A novel fractal image watermarking. IEEE Trans Multimedia 2006; 8(3): 488-499. https://doi.org/10.1109/tmm.2006.870738.

[21] Daraee F, Mozaffari S. Watermarking in binary document images using fractal codes. Pattern Recognit Lett 2014; 35: 120-129. https://doi.org/10.1016/j.patrec.2013.04.022.

[22] Lu J, Zou YR, Yang CY, Wang LJ. A robust fractal color image watermarking algorithm. Math Probl Eng 2014: 638174. https://doi.org/10.1155/2014/638174.

[23] Chung KH, Fung YH, Chan YH. Image enlargement using fractal. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing 2003; VI-273: 273-275. https://doi.org/10.1109/icassp.2003.1201671.

[24] Lai CM, Lam KM, Chan YH, Siu WC. An efficient fractal based algorithm for image magnification. In: Proceedings of International Symposium on Intelligent Multimedia, Video and Speech Processing, Hong Kong, 2004; ii-iii: 571-574. https://doi.org/10.1109/ISIMP.2004.1433972.

[25] Chen ZP, Ye ZL, Wang SX, Peng GH. Image magnification based on similarity analogy. Chaos Solitons & Fractals 2009; 40(5): 2370-2375. https://doi.org/10.1016/j.chaos.2007.10.031.

[26] Wee YC, Shin HJ. A novel fast fractal super resolution technique. IEEE Trans Consumer Electron 2010; 56(3): 1537-1541. https://doi.org/10.1109/tce.2010.5606294.

[27] Hua Z, Zhang HC, Li JJ. Image super resolution using fractal coding and residual network. Complexity 2019; 9419107. https://doi.org/10.1155/2019/9419107.

[28] Liu S, Bai WL, Zeng NY, Wang SH. A fast fractal based compression for MRI images. IEEE Access 2019; 7: 62412-62420. https://doi.org/10.1109/access.2019.2916934.

[29] Abdullahi SM, Wang HX, Li T. Fractal coding-based robust and alignment-free fingerprint image hashing. IEEE Trans Inform Forensic Secur 2020; 15: 2587-2601. https://doi.org/10.1109/tifs.2020.2971142.

[30] Khelaifi F, He HJ. Perceptual image hashing based on structural fractal features of image coding and ring partition. Multimed Tools Appl 2020; 79(27-28): 19025-19044. https://doi.org/10.1007/s11042-020-08619-w.

[31] Bisogni C, Nappi M, Pero C, Ricciardi S. FASHE: A fractal based strategy for head pose estimation. IEEE Trans on Image Process 2021; 30: 3192-3203. https://doi.org/10.1109/TIP.2021.3059409.

[32] Bisogni C, Nappi M, Pero C, Ricciardi S. PIFS scheme for head pose estimation aimed at faster face

recognition. IEEE Trans on Biometrics, Behavior, and Identity Science 2022; 4: 173-184. https://doi.org/10.1109/TBIOM.2021.3122307.

[33] Ghazel M, Freeman GH, Vrscay ER. Fractal image denoising. IEEE Trans Image Process 2003; 12(12): 1560-1578. https://doi.org/10.1109/TIP.2003.818038

[34] Ghazel M, Freeman GH, Vrscay ER. Fractal-wavelet image denoising revisited. IEEE Trans Image Process 2006; 15(9): 2669-2675. https://doi.org/10.1109/tip.2006.877377.

[35] Lu J, Ye ZX, Zou YR, Ye RS. An enhanced fractal image denoising algorithm. Chaos Solitons & Fractals 2008; 38(4): 1054-1064. https://doi.org/10.1016/j.chaos.2007.06.048.

[36] Jeng JH, Tseng CC, Hsieh JG. Study on huber fractal image compression. IEEE Trans Image Process 2009; 18(5): 995-1003. https://doi.org/10.1109/tip.2009.2013080.

[37] Lu J, Ye ZX, Zou YY. Huber fractal image coding based on a fitting plane. IEEE Trans Image Process 2013; 22(1): 134-145. https://doi.org/10.1109/tip.2012.2215619.

[38] Zou YY, Hu HX, Lu J, Liu XX, Jiang QT, Song GH. A nonlocal low-rank regularization method for fractal image coding. Fractals 2021; 29(5): 2150125. https://doi.org/10.1142/s0218348x21501255.

[39] Xu C, Ye YT, Hu ZW, Zou YR, Shen LX, Liu XX, Liu J. A primal-dual algorithm for robust fractal image coding. Fractals 2019; 27(7): 1950119. https://doi.org/10.1142/s0218348x19501196.

[40] Pan H, Liang ZY, Lu J, Tu K, Xie N. Nonlocal low rank regularization method for fractal image coding under salt-and pepper noise. Fractals 2023; 31(7): 2350076. https://doi.org/10.1142/S0218348X23500767.

[41] Liang WJ, Li XY, Tu ZH, Lu J. Multiparent fractal image coding-based methods for salt-and-pepper noise removal. Fractals 2024; 32(1): 2450012. https://doi.org/10.1142/S0218348X24500129.

[42] Wang J., Zhang M., Tong X., Wang Z.: A chaos-based image compression and encryption scheme using fractal coding and adaptive-thresholding sparsification. Phys. Scr. 97(2022) 10521. https://doi.org/10.1088/1402-4896/ac8b41.

[43] Liu, X., Tong, X., Zhang, M. et al. Image compression and encryption algorithm based on uniform non-degeneracy chaotic system and fractal coding. Nonlinear Dyn. 111, 8771-8798 (2023). https://dio.org/10.1007/s11071-023-08281-5.

[44] Long, B., Chen, Z., Liu T. et al. Improved fractal coding and hyperchaotic system for lossless image compression and encryption. Nonlinear Dyn. (2024). https://doi.org/10.1007/s11071-024-10671-2.

[45] Moon Y.H., Kim, J.S., Kim, J.H.: a fast fractal decoding algorithm based on the selection of an initial image. IEEE Trans Image Process. 5 (2000), 941-945. https://doi.org/10.1109/83.841539.

[46] Huang X.Y., He C.J., Huang H.M.: Improved scheme for fast fractal decoding algorithm based on initial image selection. Dynamics of Continuous Discrete and Impulsive Systems-Series B-Applications & Algorithms. 2 (2005), 625-629.

[47] Wang Q., Bi S., Liang D.Q.: A novel selection of the initial image for fast fractal decoding. ICIC Express Lett. Part B Appl. 4 (2013), no.5, 1253-1258

[48] Kang, H.S., Kim, S.D.: Fractal decoding algorithm for fast convergence. Opt. Eng. 35(1996), no.11, 3191-3198. https://doi.org/10.1117/1.601058.

[49] Wang Q.: An improved fast fractal decoding method. ICIC Express Lett. Part B Appl. 7 (2016), no.9, 1951-1956. https://doi.org/10.2507/icicelb.07.09.1951.

[50] Wang Q.: Accelerated decoding method in fractal image coding. Electron. Lett. 60 (2024), no.24, 1-5. https://doi.org/10.1049/ell2.70120.